194-813-SI4-005

EOSDIS Core System Project

Command Management Prototype Study Phase Results Report for the ECS Project

June 1994

Hughes Applied Information Systems
Landover, Maryland

Command Mangagement Prototype Study Phase Results Report for the ECS Project

June 1994

Prepared Under Contract NAS5-60000

APPROVED BY

Cal Moore /s/	6/20/94
Cal Moore, FOS Manager EOSDIS Core System Project	Date

Hughes Applied Information Systems
Landover, Maryland

This page intentionally left blank.

Preface

This document contains the prototype results for the study phase of the Command Management prototype, which was performed between January 1994 and May 1994. Command Management is part of the Flight Operations Segment within the EOSDIS Core System project.

This document is an informal document that is approved at the ECS Segment Manager level and does not require Government approval. After this prototype has been completed, a final report will be written.

For additional information,pertaining to the Command Management prototype, contact Jon Kuntz, FOS Offline Section Supervisor at 301-925-0622 or via electronic mail jkuntz@eos.hitc.com.

This page intentionally left blank.

Contents

Table

Abbreviations and Acronyms

1. Introduction

1.1 Purpose

This document is the first delivery of the Command Management System (CMS) Prototype Report. It documents the results of the study phase of the CMS prototype and discusses plans for the Phase I and Phase II Prototypes. The study phase focused on examining existing Command Management systems. This report documents the results of the study to date, but the study of existing systems will continue throughout the prototyping effort. Additional study results will be documented in future versions of this report.

The Command Management System (CMS) supports safe commanding of the EOS spacecraft and their associated instruments. The CMS prototype is one of four prototypes planned to explore various aspects of the FOS. The CMS prototype will help refine the CMS requirements and operations concept. The study phase began in December, 1993, and the prototyping effort is expected to continue through May, 1995.

1.2 Organization

This paper is organized as follows:

Section 1	Introduction - information about this report
Section 2	Overview - description of EOC CMS functions and CMS prototype
Section 3	Study Phase - objectives, results and current status of study
Section 4	Phase I Prototype - objectives, approach, and schedule
Section 5	Phase II Prototype - plans and schedule
	Abbreviations and Acronyms

2. Overview

2.1 EOC Command Management System

The CMS, working in concert with the Planning and Scheduling (P&S) Service, is responsible for the planned operations of the EOS spacecraft. An integrated, conflict-free Detailed Activity Schedule (DAS) for each EOS spacecraft for each target day is generated by P&S. The CMS expands the activities in the DAS to create the integrated load of spacecraft and instrument commands which will control activities on the spacecraft during the target day, and the ground script which will control real-time operations in the EOC during the target day. For the AM-1 spacecraft, the integrated load created in the EOC will be an Absolute Time Command (ATC) load consisting of stored commands to be loaded into the Spacecraft Controls Computer (SCC) stored command table. The ground script will consist of directives that control real-time commanding and telemetry monitoring software.

The CMS provides for the generation of Relative Time Sequence (RTS) and table loads using table definitions in the data base and user-provided table edits, and of ephemeris table loads containing data provided by the Flight Dynamics Facility (FDF). The CMS also accepts, validates and schedules the uplink of externally-generated loads, and validates preplanned command procedures.

2.2 CMS Prototype

The FOS operations concept calls for increased automation of control center functions. The FOS will provide this by controlling and coordinating spacecraft and ground activities based on a schedule that is generated by instrument and spacecraft engineers using a distributed P&S service. The CMS will generate the executable forms of this schedule, the ATC load and the ground script. The CMS will work in concert with P&S to validate scheduling inputs from the instrument and spacecraft engineers. The goal of the CMS prototyping effort is to reduce the risk associated with integrating the CMS into the FOS operations concept. This will be accomplished by: identification of heritage software and/or proven command management concepts to be incorporated in the operational system; improved understanding of operator interaction with the CMS resulting in a refined CMS operations concept; identification of hardware resources necessary to meet response time requirements; clarification of detailed requirements for TOO processing; and demonstration of the usefulness of object-oriented (OO) methodology in developing an FOS subsystem.

Development of the CMS prototype is following a phased approach. Initially, a study of existing command management systems was performed in order to explore and document lessons learned. The findings of this study are presented in this document. During the study phase of the prototyping effort, it was decided to use the CMS prototype as an opportunity to evaluate using a formalized OO methodology when developing an FOS subsystem. Plans for the use of this methodology are also presented in this document.

In the second phase, a working prototype of the generation of the ATC load and the ground script from the Detailed Activity Schedule will be developed. Prototyping this function will integrate the CMS, which generates the ATC load and ground script from the DAS, with the functions of P&S, which generates the DAS, and the real-time FOS functions, which are controlled by the ground script and which are responsible for the uplink of the load. The prototyping of this function will include producing OO documentation for NASA review. The results of this phase will be presented at a Prototyping Results Review as well as in the second delivery of the CMS prototype report.

Finally, in the third phase of the prototype effort, approaches to TOO support will be explored. The effort will focus on: identifying the most effective and efficient means of performing load modifications; insuring robustness; and further defining the CMS interface with the planning and scheduling subsystem. This prototype will allow refinement of the CMS specific performance requirements for responding to TOOs and late changes. The results of this phase will be presented at a Prototyping Results Review as well as in the final CMS prototype report.

Several criteria will be examined as part of the evaluation of the CMS prototyping effort. The final working prototype will be demonstrated and responses from the users will be compiled. The derived results, which will be included in the final report, will include hardware requirements and recommendations as well as COTS software recommendations. Additionally, the feedback from the demonstration will be used to modify the operational concept and detailed requirements for the CMS with regard to its functionality and ease of use. Comments on the OO documentation will be used to improve the FOS design documentation throughout the FOS development.

3. Study Phase

3.1 Objectives of Study Phase

The main objective of the study phase was to explore and document lessons learned, particularly in the areas of operational use of command management systems and evolvability of current CMS concepts and design. Secondary objectives included the identification of heritage software to be used in the FOS and the refining of the interface with the Planning and Scheduling (P&S) Service. The additional objective of training CMS development personnel in object-oriented techniques was added after the start of the study phase when formal training became available.

3.2 Results of Study Phase

The study phase included examination of existing CMS systems, informal meetings to discuss interfaces for the prototype, formal instruction in object-oriented techniques, and informal consultations with experts in the use of object-oriented techniques.

3.2.1 Command Management Systems Studied

The study phase included examination of CMSs developed or under development for NASA at GSFC and for the NOAA facility at Suitland, MD. This examination consisted of reading documentation, attending demonstrations, and interviewing CMS developers and users. Table 3-1 lists the CMSs investigated and the current status of the investigation. The subsequent sections discuss our findings to date. Some of the missions will be studied further when additional information is available, and certain aspects of some of the missions warrant more detailed investigation. The study of existing CMSs will continue throughout the CMS prototyping effort and this report will be updated accordingly.

Table 3-1. Command Management Systems Investigated (1 of 2)

Mission	Sources of information	Status
HST	Interview FOT PASS SRS (particularly Mission Scheduler, Command Loader, and Data Management functions) ICD defining PDB	Not complete. Sufficient documentation has not yet been obtained. Study will be completed when documentation is available.
UARS	CMS SRS, CMS FOT/MPG UG DFCD for CMS RDB CMS/RAC ICD ICD defining PDB CMS/FDF ICD Input Request Syntax UG Interview FOT (MMC)	All documentation obtained. Attended discussion with UARS FOT on Planning & Scheduling. Definition of activity and command data bases will be investigated further.
EUVE (EP)	CMS SRS, SUG CMS/FDF ICD Interview FOT (Loral) Sample activity plan and memory map	All documentation obtained. Interviewed FOT and attended demo of CMS and Planning systems.
SAMPEX (SMEX)	CMS SRS, SUG, DDS CMS/FDF ICD Lessons learned document Interview FOT (ATSC)	All documentation obtained. FOT interview to be scheduled.
Wind/Polar (ISTP)	CMS SRS, SUG, DDS ICD defining PDB SRR presentation ICD with POCC ICD with RUST	All documentation obtained.
FAST (SMEX)	CMS SRS, SUG, DDS CMS/FDF ICD Demo SAMPEX/FAST Command Editor	All documentation obtained. Demo of Command Editor to be scheduled.
SOHO (ISTP)	CMS SRS, SUG, DDS CMS/FDF ICD SRR presentation	All documentation obtained.

Table 3-1. Command Management Systems Investigated (2 of 2)

Mission	Sources of information	Status
SWAS (SMEX)	CMS SRS (additional doc if available)	System under development. Documentation not available. Similar to other SMEX systems.
XTE	MOC SRS ICD defining PDB MOC/FDF ICD	System under development. Documentation not available. To be studied later.
TRMM	MOC SRS (additional doc if available)	System under development. Documentation not available. To be studied later.
GIMTACS (Geosync)	GIMTACS Final Design Review GIMTACS Overview & Config. Manual GIMTACS Software Requirements GIMTACS Productivity & Reuse Case Study GIMTACS Spacecraft Schedule Generation & Maintenance Course Functional Specifications for GIMTACS FOT interview: Bob MacIntosh Developer interview: Marti Roberts	Documents still under study. Demos to be arranged.
PACS (Polar)	System Requirements Spec for PACS PACS Preliminary Design Review PACS Critical Design Review PACS System Operations Training Course PACS Ephemeris & Scheduling User's Guide Sample ground script and cmd PDB FOT interview: Chuck Liddick	Documents still under study. Demos to be arranged.
NTT NSTAR (Geosync)	Docs available Interviews: Brian Smith Dave Bryant	System under development. Documentation not available. To be studied later.

3.2.1.1 UARS

The UARS command management system encompasses the functions of planning and scheduling and load generation. A daily activity plan is developed and run against an event pool to produce an expanded activity list. An event pool consists of a set of events such as TDRSS view periods, ground contacts, and special events, and the times associated with the events. The activity list is expanded by the use of activity definitions into a command list. The command list is used to generate the command loads for the spacecraft and instruments. Most of these activities rely heavily upon the operators intervention or initiation. The FOS CMS prototype should automate the activity expansion and load generation while providing the operator with the appropriate overrides.

The concept of an activity definition is very useful. An activity definition allows the logical grouping of the commands necessary to accomplish an event. The concept could be expanded to include any ground commands necessary to support the activity on the spacecraft. When the activity is expanded, the spacecraft commands would be placed in the command load and the ground commands into the ground script.

Another useful concept found within UARS is the relative time sequence (RTS) catalog. The catalog is a repository of all of the RTSs that have been uplinked to the spacecraft. If an RTS is needed that is not currently available on the spacecraft, the CMS system retrieves the sequence from the catalog and uplinks it to the spacecraft.

The UARS command management system was developed in Fortran on mainframes and mini computers using structured analysis and design. As a result, it is not practical to use any UARS code as heritage.

3.2.1.2 EUVE

The EUVE Command management system encompasses the functions of planning and scheduling and load generation. The operator generates an activity time line. The command loads are generated under the operator's direction by using the time line, command requests, and other data. The loads are sent to the POCC for uplink to the spacecraft. The whole process relies heavily upon interaction with an operator.

During an interview with the operators, they pointed out several improvements that they would like to see in the EUVE CMS. The first improvement is the automatic inclusion of ephemeris and attitude control information into the loads. The operators are currently responsible for manually entering this information. The second improvement is an automated paper trail. This would be used to electronically log information about authorization and operator instructions. The third improvement is the ability to have multiple instances of the same process running on different workstations at the same time. Currently, only one instance of the build process is allowed to be active on any of the workstations at a given time. The final improvement is the automation of rescheduling and organizing loads after a TOO or a schedule interruption. Currently when a TOO occurs, it wipes out the remainder of the schedule. As a result, all activities after the TOO must be rescheduled manually.

The EUVE command management system was developed in Fortran on mainframes and mini computers using structured analysis and design. As a result, it is not practical to use any EUVE code as heritage.

3.2.1.3 SMEX (SAMPEX, FAST, SWAS)

The SMEX command management systems encompass planning and scheduling and command load generation. The command management systems use a set of triggers that are run against an event pool. A trigger consists of a condition or a time value and a set of activities or commands. The set is added to the activity plan when an event in the pool satisfies the trigger's condition or the activity period includes the trigger's time value. The resulting activity plan for a given period is expanded into a command list by the use of activity definitions. The command list is

converted into a command load that will be uplinked to the satellite. During this process, the operator is only responsible for modifying the triggers, adding events, initiating the generation of the activity plan, and specifying the load duration in the activity plan. The rest of the process is automatic.

The SMEX command management systems were created by using structured analysis and object oriented design. A large portion of their command management systems deals with the planning and scheduling of the activities for the spacecraft. For the FOS, the planning and scheduling subsystem will be based on heritage code from Hughes. As a result, only the portion of the SMEX systems dealing with load generation could be considered for reuse in the FOS, and much of the code for that function is specific to the SMEX spacecraft series. Many of the SMEX CMS design concepts, however, will be reused in the FOS. For example, the document "Lessons Learned from the Development of the SAMPEX CMS" recommends the use of class libraries for many of the basic objects such as queues and stacks. For the development of the FOS CMS prototype, the Hughes Class Libraries will be used to provide such objects.

3.2.1.4 WIND/POLAR

The WIND/POLAR command management system encompasses the functions of planning and scheduling and load generation. The command management system uses a spacecraft activity plan, biweekly science plans and an event pool to produce an expanded activity plan. A command list is generated from the expanded activity plan by referencing activity definitions. The command list is used to generate the command loads for the spacecraft and instruments.

The WIND/POLAR CMS system automatically includes FDF and ephemeris data into the loads by using the event pool. The operator has the ability to modify the data if necessary. The CMS system is responsible for the verification of authorization and syntax of preplanned real-time instrument commands. The FOS CMS prototype should have the capabilities to perform both tasks automatically with the appropriate operator overrides.

The WIND/POLAR command management system was developed using structured analysis and design. As a result, it is not practical to use any WIND/POLAR code as heritage.

3.2.1.5 SOHO

The SOHO command management system encompasses planning and scheduling and command load generation. The command management system has two ways of generating loads. The first method uses a set of triggers and an event pool to generate the command loads. The operator specifies when the triggers should be run against the event pool. At that time, the load is generated automatically. The second method generates loads from FDF, ephemeris, and other external information. Once the loads are generated, the operator must manually select the loads to be passed to the POCC. For the FOS, all loads should automatically be selected and scheduled for uplink with the operator having the option to override the selection.

The command management system also provides a scheduling tool that will generate an activity plan for necessary ground activities during the satellite's contact periods. The operator can automatically schedule loads by selecting the loads to be uplinked and a time period for the

activity plan. The scheduler will determine when the loads can be uplinked and generate the necessary ground commands. The FOS CMS should also automatically schedule the loads and generate the necessary ground commands with the operator having an override option.

The SOHO command management system is being developed in C/C++ for workstations using structured analysis and design. As a result, it is not practical to use any SOHO code as heritage.

3.2.1.6 NOAA

The NOAA systems are still being studied at this time. We are still in the process of procuring the documents and arranging demonstrations of the systems.

One feature that has been noted is the executable ground schedule or script that is produced by the GIMTACS command management system. The executable ground script was developed because the GOES satellites have no on board storage for commands. As a result, all commanding of the satellite must be done in real-time. The ground script is used to control both the satellite and the ground systems. The GIMTACS command management system generates the ground script through heavy interaction with the operators. The EOS CMS prototype should automate this process.

3.2.1.7 Conclusions

The study of existing CMSs has shown that systems developed using OO methodology are more easily reused than those developed using structured methodology. Of the systems studied, only the SMEX missions seem to have had requirements for evolvabiliy or reusability. The SAMPEX CMS was developed using structured analysis and OO design, and was reused extensively in the FAST CMS. This reuse was facilitated by the OO design. One specific recommendation from the study of the SMEX CMS is the use of a class library during development. A class library will provide the developers with a set of tested, low level utilities such as linked lists and hash tables.

The study has also provided a list of desirable capabilities for a command management system. To maximize efficiency in operations, a command management system should have the following capabilities:

- Automatic expansion of activities and generation of loads using database-defined expansion instructions.
- Automatic inclusion of ephemeris and attitude control information into loads.
- Automatic paper trail for authorizations and operator instructions.
- Automatic rescheduling of activities after a schedule interruption.
- Verification of authorization and syntax checking of preplanned real-time commands.
- Automatic scheduling of load uplink and generation of necessary ground commands with the appropriate operator overrides.
- Automatic generation of an executable ground script.
- Inclusion of ground directives in activity definitions.

- Maintenance of a catalog of relative time sequences used by the spacecraft.
- Simultaneous running of multiple instances of a CMS process.

The benefits of including the above capabilities in the FOS CMS will be evaluated during the prototype development phase.

3.2.2 CMS Prototype Interfaces

Informal meetings were held with FOS development personnel throughout the study phase to discuss interfaces between the CMS and other FOS services. As a result of these meetings, the Phase I Prototype will include initial versions of the following interfaces:

- Planning & Scheduling Service - Detailed Activity Schedule
- Real-Time Commanding Service - ATC Load
- Data Management Service - Activity Definition and Command Definition Databases
- User Interface Service - CMS User Interface, Command Procedures, Ground Script

3.2.3 Object-Oriented Techniques Studied

The software to provide CMS functions will be developed using an OO approach. During the study phase of the CMS prototype, CMS development personnel and the NASA ETM for CMS received formal instruction in OO analysis and design techniques. This instruction focused on the Object Modeling Technique (OMT) described in Object-Oriented Modeling and Design (Rumbaugh, et.al., 1991). This technique includes an object model, a dynamic model, and a functional model where appropriate. A partial CMS object model was developed as a training exercise. CMS development personnel were also trained in the C++ language.

Experts in OMT are available for consultations at the Hughes Landover facility and will remain on site through June, 1994. These experts have been consulted on the CMS object model, particularly in the area of DAS processing. CMS development personnel will continue to use this resource in developing the Phase I Prototype.

4. Phase I Prototype

4.1 Objectives of Phase I Prototype

The main objective of the Phase I Prototype is to demonstrate the generation of an ATC load which controls spacecraft activities during a target day and a ground script which controls EOC activities during the same target day from an integrated, conflict-free schedule which P&S has produced for that target day. The creation of a ground script by a CMS and its use to automatically execute repetitive FOT functions is a feature of Loral heritage systems but will be implemented for the first time at GSFC in the EOC. The creation of an ATC load by expanding the activities in an integrated, conflict-free schedule is a feature of GSFC heritage systems but will be implemented in conjunction with a distributed P&S service for the first time in the EOC. Prototyping these features will demonstrate the feasibility of integrating these features into the FOS design.

A secondary objective of the Phase I prototype is to explore the use of OMT as a documentation vehicle. Object-oriented analysis and design and the types of documentation used to explain them may be unfamiliar to the NASA and spacecraft contractor personnel who will review the FOS design. The CMS prototype will include opportunities for NASA personnel to review the OO documentation of the prototype's design.

Additional objectives of the Phase I Prototype are continued study of the requirements for the operator interface with the CMS, definition of Level 4 requirements for ATC load generation, and the identification of hardware resources necessary to meet response time requirements.

4.2 Approach to Phase I Prototype

The CMS prototype will focus on CMS interfaces with P&S, the data management subsystem, the real-time subsystem, and the user interface subsystem. Test data for the CMS prototype will include a DAS containing activities that can be expanded into the ATC load and ground script, an activity definition data base containing both spacecraft and ground type activities, a command definition data base to convert commands in the ATC load from mnemonic to binary form, and a set of command procedures that can be referenced by the ground script. Developing these test data as part of the prototyping effort will call attention to deficiencies in the current understanding of these interface objects and speed their definition.

The user interface needed to perform the CMS functions will be studied during Phase I Prototype development. The baseline staff recommendations in the FOS Operations Trade Study for the ECS Project propose that the responsibilities of the command management analyst be subsumed by the schedulers. By showing that the ATC load and ground script can be generated from the DAS, the CMS prototype will help to validate the extent to which command management functions can be automated.

The CMS Phase I Prototype will be developed using OMT methodology. The documentation produced will include: written Level 4 requirements; an object model of the DAS processing function of the CMS, including an object diagram with definitions of the classes and their attributes and behaviors; a dynamic model of the DAS processing function including a scenario, an event trace, and a state diagram; and functional models as appropriate. The requirements and design documentation will be presented at an informal design review.

The software will be developed in C++ using the Hughes Class Libraries. This will facilitate the reuse of the software developed for the prototype in the operational system.

4.3 Schedule for Phase I Prototype

Prototype Requirements Definition	5/94
Prototype Design	5/94 - 7/94
Informal Design Review	7/94
Prototype Development	5/94 - 10/94
Formal Demonstration	11/94
Prototype Results Report, Second Delivery	1/95

5. Phase II Prototype

5.1 Plans for Phase II Prototype

The Phase II Prototype will focus on approaches to TOO support. Nominally, the CMS will receive one DAS and build one integrated load per day. However, the CMS for ECS must support TOOs and late changes. The operations concept for handling TOOs and late changes continues to evolve, but the CMS will probably be required to provide updates to loads that have been generated and are awaiting uplink and/or loads that already have been uplinked. The Phase II Prototype will explore the possibilities of generating partial loads and/or load patches.

The Phase II Prototype will be built on software generated for the Phase I Prototype. The CMS interfaces with the P&S, real-time, and user interface subsystems for handling TOOs and late changes will be very similar to those required for the generation of the ATC load and ground script from the DAS.

5.2 Schedule for Phase II Prototype

Prototype Requirements Definition	12/94
Prototype Design	12/94 - 1/95
Prototype Development	12/94 - 2/95
Formal Demonstration	3/95
Prototype Results Final Report	5/95

This page intentionally left blank.

Abbreviations and Acronyms

ATC	Absolute Time Command
CMS	Command Management System
DAS	Detailed Activity Schedule
ECS	EOSDIS Core System
EOC	EOS Operations Center
EOS	Earth Observing System
EUVE	Extreme Ultraviolet Explorer
FAST	Fast Auroral Snapshot Explorer
FDF	Flight Dynamics Facility
FOS	Flight Operations Segment
FOT	Flight Operations Team
GIMTACS	GOES I-M Telemetry and Command System
OO	Object Oriented
P&S	Planning and Scheduling Service
PACS	Polar Acquisition and Control System
POCC	Payload Operations Control Center
POLAR	Polar Plasma Laboratory
RTS	Relative Time Sequence
SAMPEX	Solar Anomalous and Magnetospheric Particle Explorer
SMEX	Small Explorer
SOHO	Solar and Heliospheric Observatory
TOO	Target of Opportunity
UARS	Upper Atmosphere Research Satellite
WIND	Interplanetary Physics Laboratory